Amendments to the Claims:

This listing of claims will replace all prior versions, and listings, of claims in the application:

Listing of Claims:

Please cancel claim 12 without prejudice and amend claims 1-4, 6-11, 13-21, 23, and 42-44 as follows:

1. (currently amended): A computer implemented method to ~~transform~~ execute a code splitting program comprising the steps of:

splitting ~~the~~ a processor program into a set of control structure instructions including a selected set of address changing instructions and a set of arithmetic/logic (AL) instructions;

reducing the set of AL instructions to a reduced set of AL instructions by removing duplicate AL instructions;

assigning to each AL instruction of a first AL instruction type in the reduced set of AL instructions an address in ~~at least one~~ a first AL memory; ~~and~~

assigning to each AL instruction of a second AL instruction type in the reduced set of AL instructions an address in a second AL memory;

generating instruction fetch (IF) instructions in a sequencing order determined from the set of control structure instructions and specified by the IF instructions for programmably selecting AL instructions to be fetched from ~~said~~ at least one of the first AL memory or the second AL memory, wherein ~~the~~ a first IF ~~instructions~~ instruction type ~~have~~ has ~~an~~ a first IF instruction format which contains information to ~~identifies~~ identify for execution at least one assigned address of at least one AL instruction from the first AL memory or from the second AL memory[[,]] and a second IF instruction type has a second IF instruction format which contains information to identify for parallel execution at least one assigned address of at least one AL instruction from the first AL memory and at least one assigned address of at least one AL instruction from the second AL memory, whereby the processor program is transformed into a sequence of IF instructions and ~~a list~~ the reduced set of AL instructions at assigned addresses.

2. (currently amended): The computer implemented method of claim 1 wherein the set of AL instructions are non-control instructions comprised of a selected set of adds, subtracts, multiplies, divides, logical functions, shifts, rotates, permutations, bit operations, and other arithmetic and logic type functions.

3. (currently amended): The computer implemented method of claim 1 wherein any AL instruction of said reduced set of AL instructions stored in said ~~at least one~~first AL memory or in said second AL memory can be accessed a plurality of times without requiring the AL instruction to be duplicated in the ~~at least one~~first AL memory or in the second AL memory.

4. (currently amended): The computer implemented method of claim 1 wherein said reduced set of AL instructions is comprised of at least one ~~sequence~~ set of AL instructions that are to be executed singly.

5. (previously cancelled)

6. (currently amended): The computer implemented method of claim 1 wherein said reduced set of AL instructions is comprised of at least one ~~sequence~~ set of AL instructions which can be issued in parallel.

7. (currently amended): The computer implemented method of claim 1 wherein said set of control structure instructions are comprised of a selected set of for-do, if-then-else, case, while-do, do-until, do-while, branches, calls, returns, and auto-loop instructions.

8. (currently amended): The computer implemented method of claim 1 wherein said processor program is analyzed to identify AL instructions that are required to be executed sequentially, AL instructions that can be executed in parallel, said set of control structure instructions, and the sequence of instructions.

9. (currently amended): The computer implemented method of claim 8 wherein the identified instructions and the sequence of instructions are used to generate the IF instructions in ~~a~~the sequencing order, the sequencing order controlled by information contained in the IF instructions, wherein the IF instructions are of a different type and format than instructions used in the program.

10. (currently amended): The computer implemented method of claim 1 wherein said reduced set of AL instructions are split into at least two sets of AL instructions with an AL instruction from each set assigned an address from one of at least two AL memories.

11. (currently amended): The computer implemented method of claim 1 further comprises the steps of:

identifying single and duplicate AL instructions in a section of code making up said program;

removing all but one of the duplicate AL instructions from said at least one AL memory, whereby the remaining one of the duplicate AL instructions is a single reference AL instruction;

identifying at what addresses in the program sequence the duplicate AL instructions occur and the address of the single reference AL instruction; and

generating IF instructions in a sequencing such that whenever a duplicate AL instruction is required, an IF instruction is executed to create the address for the single reference AL instruction stored in said at least one AL memory, whereby a single AL instruction is stored in said ~~at least one~~first AL memory or in said second AL memory instead of a plurality of duplicate AL instructions for said section of code making up said program.

12. (canceled)

13. (currently amended): The processor system of claim ~~12~~ 17wherein ~~a~~ the first IF instruction type comprises an opcode field specifying a sequential fetch operation and an IMemory address field, and a ~~second~~ third IF instruction type comprises the fields of said first IF instruction plus an additional field indicating the number of instructions to be sequentially fetched and specifying the IMemory address as the starting address for a group of instructions.

14. (currently amended): The processor system of claim 13 wherein the first and ~~second~~ third IF instruction types each comprise an additional field for an IF memory instruction address.

15. (currently amended): The processor system of claim 13 wherein the first and ~~second~~ third IF instruction types each comprise at least one additional field for a conditional branch address specifying a location in the IF memory.

16. (currently amended): The processor system of claim 13 wherein the first and ~~second~~ third IF instruction types each comprise two additional fields for a loop count and a loop

end address; the address of the IF instruction identifying the loop start address which together with said loop end address identifies the program loop and the loop count controls the number of iterations of the loop.

17. (currently amended): ~~The~~ A processor system ~~of claim 12 further~~ comprising:

a code splitting tool for transforming a program by generating an instruction addressing control program as a sequence of instruction fetch (IF) instructions and at least one set of non-control instructions;

an instruction fetch (IF) memory storing the sequence of IF instructions;

a programmable instruction fetch mechanism that is programmed by the IF instructions to fetch and execute IF instructions in a sequencing order, wherein the sequencing order is controlled by information contained in each of the IF instructions;

at least one non-control instruction memory (IMemory) storing the at least one set of non-control instructions, whereby an IF instruction is formatted as a first IF instruction type to identify at least one address of the at least one IMemory and said programmable instruction fetch mechanism operates to fetch IF instructions from said IF memory and execute each fetched first IF instruction type to generate at least one IMemory instruction address to select at least one non-control instruction to be fetched from the at least one IMemory for execution;

a ~~third~~ second IF instruction type for parallel multiple-issue instructions; and

an additional non-control instruction memory (IMemory) comprising a second set of non-control instructions, ~~whereby~~ said programmable instruction fetch mechanism operating to fetch IF instructions from said IF memory and when executing ~~a third~~ the second IF instruction type generating at least two IMemory instruction addresses to select non-control instructions to be fetched from the at least two IMemories for execution in parallel.

18. (currently amended): The processor system of claim 17 wherein said ~~third~~ second IF instruction type comprises at least three fields: an opcode field specifying a parallel multiple-issue fetch operation, and at least two IMemory addresses.

19. (currently amended): The processor system of claim 18 wherein the ~~third~~ second IF instruction type comprises an additional field for an IF memory instruction address.

20. (currently amended): The processor system of claim 17 wherein the ~~third~~ second IF instruction type comprises at least one additional field for a conditional branch address specifying a location in the IF memory.

21. (currently amended): The processor system of claim 17 wherein the ~~third~~ second IF instruction type comprises two additional fields for a loop count and a loop end address; the address of the IF instruction identifies the loop start address which together with said loop end address identifies the program loop and the loop count controls the number of iterations of the loop.

22. (original): The processor system of claim 17 wherein a fourth IF instruction type comprises at least five fields:

a load IMEM instruction opcode;

at least two base address register indicator bits;

at least one IMemory offset; and

at least one data memory offset.

23. (currently amended): The processor system of claim ~~12~~ 17 further comprising:

a fifth IF instruction type for parallel multiple-issue instructions;

a second non-control instruction memory (IMemory) comprising a second set of non-control instructions; and

a third non-control IMemory comprising a third set of non-control instructions , said programmable instruction fetch mechanism operating to fetch IF instructions from said IF memory and when executing a fifth IF instruction type generates at least three IMemory instruction addresses to select instructions to be fetched from the at least three IMemories for execution in parallel.

24. (original): The processor system of claim 23 wherein the fifth instruction type comprises at least three fields:

an opcode field indicating a multiple instruction fetch operation;

at least one IMemory address field specifying a common address for the at least two IMemories; and

a separate IMemory address field specifying an address for a separate IMemory.

25. (previously cancelled)

26. (previously presented): The processor system of claim 27 further comprising operation steps to select at least two PE AL instructions for execution on the at least two AL decode and execute units.

27. (previously presented): A processor system comprising:

an instruction fetch (IF) memory comprising a sequence of IF instructions;

a programmable instruction fetch mechanism comprising means to fetch and execute IF instructions;

at least two IMemory address bus interfaces between the programmable instruction fetch mechanism and at least one processing element (PE);

at least one PE further comprising:

at least two arithmetic/logic (AL) instruction memories (IMemories) which interface with the at least two IMemory address buses;

at least two AL decode and execute units; and

a set of address registers to be used for addressing operations, wherein said programmable instruction fetch mechanism operating to fetch IF instructions from said IF memory and executes the fetched IF instructions thereby generating IMemory instruction addresses to select processor element (PE) AL instructions singly from one of the AL instruction memories for execution on one of the AL decode and execute units[[.]]; and

at least two processor elements controllable as one concatenated processor element with a first type IMemory AL instruction specifying a concatenated operation, and controllable as two independent processor elements with a second type IMemory AL instruction specifying at least two independent operations.

28. (original): The processor system of claim 27 wherein the second type IMemory AL instruction is a subset of the first type IMemory AL instruction.

29. (previously presented): The processor system of claim 27 wherein the PE AL instructions comprise multiple PE AL instruction formats including an optional vector parameter field and an optional conditional execution field.

30. (previously presented): The processor system of claim 27 wherein a PE AL instruction format comprises an opcode field, a data type field, a data memory selection field, and at least one operand field.

31. (previously presented): The processor system of claim 27 wherein a PE AL instruction format comprises an opcode field, a vector parameter field, a vector address register field, a data memory selection field, and at least one operand field.

32. (previously presented): The processor system of claim 27 wherein a PE AL instruction format comprises an opcode field, a vector parameter field, a data type field, a data memory selection field, and at least one vector operand parameter field, at least one address register field, and at least one operand offset field.

33. (previously presented): The processor system of claim 27 wherein a vector operation executes on at least one PE, the system further comprising an execution sequence of initiating a vector operation when a PE AL instruction format supporting vector setup operation executes, causing at least one operand address to be loaded into an address register and starting the vector operation each time a PE AL instruction format supporting vector system is fetched and executed.

34. (previously presented): The processor system of claim 27 wherein a PE AL instruction format comprises an opcode field, a data type field, a data memory selection field, at least one operand field, and a 16-bit immediate field.

35. (previously presented): The processor system of claim 27 wherein a PE AL instruction format comprises an opcode field, a data type field, a data memory selection field, at least one operand field, and a 32-bit immediate field.

36. (previously presented): The processor system of claim 27 wherein the PE AL instructions of a first IMemory AL instruction type are comprised of multiple 64-bits formats.

37. (original): The processor system of claim 36 wherein the PE AL instructions of a second IMemory AL instruction type are comprised of multiple 32-bit formats.

38. (original): The processor system of claim 37 wherein the PE AL instructions of a third IMemory AL instruction type are comprised of multiple 16-bit formats.

39. (original): The processor system of claim 38 wherein the 16-bit format PE instructions comprises a target register specified as a function of one of the source operand fields.

40. (previously presented): The processor system of claim 27 further comprising at least two clusters of two processor elements each controllable as two processor elements with two different first type IMemory AL instructions, and controllable as four independent processor elements with four different second type IMemory AL instructions.

41. (previously presented): The processor system of claim 27 further comprising at least two clusters of four processor elements each controllable as two processor elements with two different first type IMemory AL instructions, controllable as four independent processor

elements with four different second type IMemory AL instructions, and controllable as eight independent processor elements with eight different second type IMemory AL instructions.

42. (currently amended): A processor system comprising:

a code splitting tool for transforming a program by generating an instruction addressing control program as a sequence of instruction fetch (IF) instructions and at least one list of non-control instructions, wherein the IF instructions were not used in the program;

an instruction fetch (IF) memory storing the sequence of IF instructions;

a programmable instruction fetch mechanism that is programmed by the IF instructions to fetch and execute IF instructions in a sequencing order, wherein the sequencing order is controlled by information contained in each of the IF instructions;

at least ~~one~~ two non-control instruction ~~memory~~ memories (~~IMemory~~IMemories) each storing a set of non-control instructions, whereby an IF instruction is formatted to identify at least ~~one~~ two addresses of the at least ~~one~~ two ~~IMemory~~ IMemories and said programmable instruction fetch mechanism operating to fetch IF instructions from said IF memory and execute ~~each~~ at least one fetched IF instruction to generate at least ~~one~~ two IMemory instruction addresses to select at least ~~one~~ two non-control instruction to be fetched from the at least ~~one~~ two ~~IMemory~~ IMemories for execution and to ~~generate~~ identify an address for the next IF instruction.

43. (currently amended): The computer implemented method of claim 1 further comprising:

storing the IF instructions in an IF memory; and storing the reduced set of AL instructions in the ~~at least one~~ first AL memory.

44. (currently amended): The computer implemented method of claim 43 further comprising:

fetching in the sequencing order the IF instructions from the IF memory;

executing the fetched IF instructions, whereby the execution of each fetched IF instruction generates at least one AL memory address;

fetching ~~the~~ AL instructions from said ~~at least one~~ first AL memory at the AL memory addresses generated by executing the fetched IF instructions; and

executing the fetched AL instructions, whereby the function of said processor program is accomplished.